Caratteristiche web service ServizioFidelity

1. Definizione di richiamo nel codice

Url di posizionamento: https://www.fidelity4web.com/services/serviziofidelity.asmx

password soapheader: fidelity4WebWebService!

p.s: lanciando dal browser l'indirizzo di posizionamento del webservice è possibile visualizzare l'elenco delle funzioni e, cliccando su ognuna, ottenere la definizione soap e le caratteristiche xml dei parametri e delle risposte.

2. Esempio richiamo di metodo

Per gli esempi di codice dimostrativi vi rimandiamo alla pagina https://www.tesisinformatica.com/it/fidelity-web/api , dove è possibile scaricare alcuni progetti demo sviluppati in diversi linguaggi di programmazione nei quali si richiama il servizio web. I linguaggi dei progetti demo attualmente disponibili sono Vb.net, C#, C++, Asp.net Vb.net, Asp.net C# e Php.

La struttura comune a tutti i linguaggi è la seguente:

- Oggetto istanza alla classe riferita al servizio web (identificata per comodità come wsFidelity);
- Oggetto istanza alla classe SoapHeaderData;
- Password della classe SoapHeaderData;
- Funzione.

3. Strutture ed Enumerazioni

- **Premio (Classe)**

```
idPremio Integer
nome String
punti Decimal
```

- **tipoMov (struttura)**

```
spesa Integer
prepagato Integer
ricarica Integer
```

- **tipologie (enumerazione)**

```
spesa = 1
ricarica = 2
prepagato = 3
premio = 4
```

- **resocontoSpesaEsteso (struttura)**

```
punti Decimal
credito Decimal
eseguito Boolean
idMovimento Integer
```

- **datiClienteEsteso (struttura)**

```
nome String
punti Decimal
credito Decimal
bloccato Boolean
```

4. Elenco metodi e funzioni

- `assegnaPremio(codGestore Integer, nCard String, idPremio Integer) resocontoSpesaEsteso`

  Funzione che permette di registrare l'assegnazione di un premio ed il relativo scarico di punti. La risposta ricevuta sarà una struttura che conterrà, nel caso sia andato tutto a buon fine, l'id del movimento(indicato come `idMovimento`) utile nel caso in cui lo si dovesse successivamente stornare. Il valore booleano `eseguito` darà la conferma o meno che l'operazione sia stata svolta correttamente. In caso di errore invece `idMovimento` verrà restituito uguale a zero.
  Gli altri valori restituiti, ossia `punti` e `credito`, indicheranno la situazione attuale del tesserato.

- `caricaDatiCliente(nCard String) datiClienteEsteso`

  Funzione che permette di caricare i dati del cliente dopo aver indicato come parametro il numero della card. Nel caso in cui la card non sia registrata nel database viene restituita un'eccezione. La risposta viene fornita sotto forma di struttura contenente alcuni dati relativi all'intestatario della tessera e i dati della sua fidelizzazione (per dettagli maggiori vedere la composizione della struttura `datiClienteEsteso` al punto 3).

- `caricaIdGestore(user String, password String) Integer`

  Funzione che permette di recuperare un numero intero corrispondente all'id del gestore all'interno del database fornendo come parametro user e password dello store. I dati passati come parametro corrispondono alle credenziali fornite allo store per l'accesso al pannello dedicato nel sistema Fidelity Web 2.0.

- `caricaIdTesserato(nCard String) Integer`

  Funzione che permette di recuperare un numero intero corrispondente all'id del tesserato all'interno del database fornendo come parametro il numero della card.

- `elencoPremiRaggiunti(nCard String)List(Of premio)`

  Funzione che permette di ottenere l'elenco dei premi raggiunti dalla card inserita come parametro. La risposta da parte del servizio sarà strutturata come elenco di oggetti `premio` (per dettagli maggiori vedere la composizione della classe `premio` al punto 3). Nel caso in cui non ci siano premi riscattabili da parte del tesserato la lista viene restituita vuota.

- `esisteCard(nCard String) Boolean`
  Funzione che permette di testare se il numero della card passata come parametro corrisponde ad una tessera registrata nel database. Il valore booleano restituito indicare se la card è già presente all'interno dell'archivio dati o se figura come nuova.

- `HelloWorld () String`
  Funzione che testa la corretta connessione al servizio web . Restituisce una stringa contenente un testo di benvenuto, utile per verificare il corretto aggancio del web service all'interno del codice.

- `inserisciCliente(codGestore Integer, nCard String, cognome String, nome String, dataNascita DateTime, sesso String, cf String , indirizzo String, cap String, citta String, prov String, cellulare String, email String, nCardPresentante String) Boolean`
  Funzione che permette di registrare un nuovo cliente indicando lo store che lo registra e i dati primari. E' possibile anche inserire il nuovo cliente indicando solamente il numero della card. La funzione restituisce una valore booleano di conferma al termine dell'operazione.

- `inserisciMovimento(codGestore Integer, nCard String, importo Decimal, tipoMovimento tipologie) resocontoSpesaEsteso`
  Funzione che permette di registrare un movimento indicando lo store che lo esegue,la card del cliente, l'importo di spesa e la tipologia del movimento (`tipoMovimento` è un enumerazione, quindi nel caso si voglia registrare una spesa si dovrà passare come parametro 1 oppure spesa). La risposta ricevuta sarà una struttura che conterrà, nel caso sia andato tutto a buon fine, l'id del movimento(indicato come `idMovimento`) utile nel caso in cui lo si dovesse successivamente stornare. Il valore booleano `eseguito` darà la conferma o meno che l'operazione sia stata svolta correttamente. In caso di errore invece `idMovimento` verrà restituito uguale a zero.
  Gli altri valori restituiti, ossia `punti` e `credito`, indicheranno la situazione attuale del tesserato.

- `isCardValida(nCard String) Boolean`
  Funzione che permette di testare se il numero della card passata come parametro è un codice valido all'interno del sistema di fidelizzazione. Il valore booleano restituito indicare se il codice sia valido o meno.

- `sostituisciCard(codGestore Integer, nCardAttuale String, ncardNuova String) Boolean`
  Funzione che permette di sostituire la card ad un cliente indicando lo store che esegue l'operazione, la card attuale e il numero della nuova card.

- `stornaMovimento(codGestore Integer, idMovimento Integer) Boolean`
  Funzione che permette di stornare un movimento(se stornabile) e restituisce un valore booleano che indica se l'operazione è stata eseguita o no. La funzione è valida anche per lo storno dei premi assegnati. I parametri necessari per sfruttare questa funzione sono il codice identificativo dello store e l'id del movimento.

- **`tipoMovimento()`** `TipoMov`
  Funzione che restituisce l'elenco completo delle tipologie di movimento registrabili, indicando per ogni operazione disponibile il numero intero identificativo (per dettagli maggiori vedere la composizione della struttura **`TipoMov`** al punto 3).